T-76.4115
Software Development Project

“GUI for Game on Demand Service with PHP”

Group 11 – Game On! Solutions

I2 Iteration Plan

Table of Contents

Change log:

14.1.2007	Initial version of this document / IL
21.1.2007	Updated section 4 / IL
28.1.2007	Updated sections 3.3 and 4 / IL
26.2.2007	Updated sections 3.3 and 4 / IL

1. Background

Game On! Solutions signed on to a project for G-cluster to develop a GUI for Game on Demand Service with PHP. This document defines the goals, tasks and schedule of the third iteration.

The content of this document has been agreed upon with the customer in meeting which was held 11.1.

2. Schedule

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
3	15.1. -Begin of I2	16.1. -Return I2 iteration plan -Layout mock-ups from graphics designer	17.1. -Multiplayer functionalities	18.1.	19.1. -Support for skins -Profiles & selection -User management	20.1. -Continuous integration	21.1. -Game session data -Implementing new layout
4	22.1. -1st round of usability testing	23.1. -Layout design	24.1. -Buddy list -Unit test for old components -High scores -Sales admin	25.1.	26.1. -Code review -Advertisements	27.1.	28.1. -Multiplayer lobby -Messaging
5	29.1. -Implementing graphics -Functional freeze	30.1.	31.1. -Setting up demo environment -Commit freeze	1.2. -Demo to customer -Freezes over	2.2. -Separate nick for each game -2nd round of usability testing	3.2.	4.2. -Multiplayer room -Tournaments -Game filters -Statistics

	Mon	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**	**Sun**
6	5.2.	6.2.	7.2.	8.2. -Code review -Static analysis	9.2. -Usability testing with peer group	10.2. -Writing test cases -Front-end tests	11.2. -Billing -Credits -Functional freeze
7	12.2.	13.2.	14.2.	15.2.	16.2.	17.2. -Final system to peer group for testing	18.2.
8	19.2.	20.2.	21.2. -Report testing results to peer group	22.2.	23.2.	24.2.	25.2. -Fixing problems found by peer group
9	26.2. -Delivery of documents -Commit freeze	27.2. -Setting up demo environment	28.2.	1.3. -End of I2			

3. Iteration planning

3.1 Goals

The primary goal of this iteration is to finish up the developed system and deliver it to the customer. This includes the creation and updating of all of the manuals and documentation.

To finish the system, more complex multiplayer functionalities will be needed and this is one goal for this iteration. Along with the multiplayer functionalities comes the functionalities which help users interact with each other.

Because this is the last development iteration of the system, one goal is to clean up the code and make sure it works properly with thorough testing

3.2 Deliverables

Document	**Assignee**	**Deadline**
Iteration plan	Ismo Lehmus	16.1.
Project plan	Ismo Lehmus	26.2.
Test cases	Juuso Jääskeläinen	26.2.
Test charters	Juuso Jääskeläinen	26.2.
Test matrix	Juuso Jääskeläinen	26.2.
QA Report	Juuso Jääskeläinen	26.2.
Progress report	Ismo Lehmus	26.2.
Testing guidelines for peer group	Juuso Jääskeläinen	16.2.
Testing results for peer group		21.2.
User manual	Antti Kaunisto	26.2.

3.3 Tasks

DL	Task	Assignee	Status	Estimated time	Spent time
16.1.	Iteration planning	Lehmus	Finished	20	11
17.1.	Listing of open multiplayer games	Kaunisto	Finished	4	5
17.1.	Multiplay now	Kaunisto	Finished	2	5
19.1.	Profiles, accounts, users	Merinen	Mostly done	5	14,7
19.1.	End-user login & profile switching	Merinen	Mostly done	2	3
19.1.	Support for skins	Leiponen	Finished	8	14
19.1.	User management (inc. general structure)	Merinen	Finished	10	20
20.1.	Continuous integration (inc. PHPdoc generation)	Jääskeläinen	Finished	6	5
21.1.	Implementing new layout	Auvinen	Finished	20	26
21.1.	Game session data saving	Pihamaa	Somewhat done	5	15,25
22.1.	Usability testing	Workgroup	Finished	10	10
23.1.	Layout design	Workgroup	Finished	16	18
24.1.	Buddy list	Merinen	Mostly done	6	7
24.1.	Unit tests for old components	Everyone	Finished	25	37,75
24.1.	High scores	Pihamaa	Somewhat done	10	11
24.1.	Sales admin	Leiponen	Mostly done	4	9
26.1.	Advertisements	Auvinen	Finished	10	12,5
28.1.	Multiplayer lobby	Kaunisto	Mostly done	20	13
28.1.	Messaging	Merinen	Mostly done	12	21
29.1.	Implementing new graphics	Auvinen	Finished	20	31
31.1.	Setting up demo environment	Kaunisto	Finished	4	3
1.2.	Demo to customer	Workgroup	Finished	6	5
2.2.	Separate nick for each game	Merinen	Finished	4	2
2.2.	Usability testing with real graphics	Workgroup	Finished	10	5
4.2.	Multiplayer rooms	Kaunisto	Mostly done	20	24
4.2.	Tournaments	Pihamaa	Somewhat done	25	15,3
4.2.	Game filters	Leiponen	Somewhat done	8	1
4.2.	Statistics	Merinen	Mostly done	4	4
8.2.	Static analysis	Jääskeläinen	Finished	1	1
9.2.	Usability testing	Workgroup	Finished	15	18,3
10.2.	Writing test cases	Jääskeläinen	Finished	4	8
10.2.	Front-end tests	Merinen	Somewhat done	5	2
11.2.	Billing	Merinen	Dropped	5	0
11.2.	Credits	Pihamaa	Dropped	6	0
17.2.	Deliver final system to peer group for testing	Lehmus	Finished	3	1
21.2.	Peer group testing, report results	Lehmus	Finished	10	7
26.2.	User manual	Kaunisto	Somewhat done	15	7
27.2.	Setting up demo environment	Kaunisto	Somewhat done	4	0
	System integration	Leiponen	Mostly done	15	6,5
	Testing	Jääskeläinen	Somewhat done	15	20,5
	Updating mandatory docs	Lehmus, Jääskeläinen	Mostly done	25	13,5

	Time tracking	Lehmus	Mostly done	15	5,5
	Generic communication	Everyone		14	44,5
	Generic studying	Everyone		6	7
	Top10 games (not planned)	Pihamaa	Mostly done	0	1,5
	Meeting with the mentor (not planned)	Everyone	Finished	0	9
	Final demo (not planned)	Everyone	Finished	0	17
	Time used on I1½ / unplanned tasks			0	28,3

3.4 Deadline specifications

The deadlines mentioned for all tasks mean that the specified day is the last one when work on the issue should be done, ie. the task must be finished by 6am the following day, unless otherwise specified. In addition to the mentioned deadlines, two other points in time are specified:

Functional freeze points : 29.1.2007 at 06:00 and 11.2. at 06:00
After this deadline no new functionalities may be added to the repository, meaning that everything must be either finished or dropped from the current iteration. Any unfinished work must be notified to the project management. Bug fixes and other improvements to otherwise finished functionalities may be committed without prior authorization by the QA manager. The function freeze point is at 06:00 in the morning, meaning that no actual work on issues can be done during this day.

Commit freeze points : 31.1.2007 at 06:00 and 26.2. at 06:00
After this point no commits can be made to the repository without an explicit prior authorization by the QA manager. There should be no need for any changes at this point. If some bug fixes are still finished at this point, they have to first be cleared with the QA manager, so that he can test the affected parts of the system after the commit. This ensures that the version checked out from the repository for the demo session works perfectly and is thoroughly tested. Other work can continue in unhindered in working copies. The commit freeze ends by 18:00 the next day, 11.1.2007.

3.5 Tasks in detail

Iteration planning

Iteration planning consist of a meeting with the customer, deciding what to do and with priority and creating and updating this document so that it can be used to steer the development to right direction.

Assignee(s): Lehmus, Jääskeläinen
Time estimate: 20 hours
Deadline: 16.1.

Profiles, accounts, users

The profile system is two-tiered. There are subscription accounts and user profiles. The accounts represent individual billing accounts and would in the end environment mostly be individual households. The profiles represent individual persons. For example, for one account there could be four profiles: Dad, mom, daughter and son.

The account also acts as a store for the billing information and is linked to the profiles.

The account number's minimum length must be modifiable from the configuration file. Default length 12 numbers and the code is always numeric. See the next task for more information.

In addition, there are administration side users, which are different from the end-user side profiles and accounts. See the user management task for more information.

Technical specification:

The data related to the different entities should at the least include the following:
Accounts:

- Account number
- Account pin code
- Credits

Profiles:

- Profile name
- Actual name (optional)
- Pin code (optional)
- Username & password – configurable on a per-deployment basis if these are shown at all
- Age
- Ability to create new profiles and to modify age (on/off)
- Avatar image id (see My profile) related to the image stored on the database. Create methods for accessing and modifying this information.

Developer(s): Merinen

Design pair: Jääskeläinen
Time estimate: 5 hours
Deadline: 17.1.

End-user login & profile switching

By logging in to an account the right to use the system is verified and the related information linked to the user.

The idea is that the account can be automatically detected in the eventual setup, thus circumventing the need for generic login. In the version currently being implemented, this is simulated by adding an ip-address field to the accounts which, at the switch of a configuration directive, will be used to detect the remote user and login her to the account automatically. Obviously, this functionality will most likely not be used in any actual circumstances because of the high (or at least currently unquantified) security risks involved, and should be labelled accordingly.

There must also be a way to circumvent the profile system by a) some technical piece of information stored in the database and provided by the end client on connect and b) by a system-wide configuration setting, specifying the only account always used for that deployment.

If no account is automatically detected, the general login screen is shown before access to the system is granted. This login screen contains boxes for both account-number based login and for username + password based login directly to a profile. This means that there are two ways to login:

1) By providing the account number and the pin code
2) providing the username and password related to the profile (logins to the profile directly, as well as to the account)

Note that administrators and salesmen cannot directly login to the end user system, since their user accounts are not related to any end-user account.

For each account there is one default profile which is used if no other profile is selected. The age of this default profile must be configurable on a per-deployment basis.

The profiles must be selectable from the front page and from the “my profile”-page. If under around 5 profiles are linked to the current account, they will all be listed on the front page. Otherwise the front page will show a link to a separate profile selection page listing all the available profiles. After a profile is selected, the pin-code/password for that profile is queried from the user, if it is set (the codes are optional).

The account login must be checked on every page load. If no account login is detected, the login page must be supplied to the user instead of the requested page. After the login, the user must be redirected to the page originally requested. The assignee of this task must also make sure that the cron-tasks, such as the metadata updates, can circumvent this authentication verification. I.e. cron tasks do not require a login and no login page is outputted to them. This part also applies to the administrator side.

Developer(s): Merinen
Design pair: Jääskeläinen
Time estimate: 2 hours
Deadline: 17.1.

Listing of open multiplayer games

Generates a list that shows all multiplayer games that are currently open (waiting for more players) by paginable pages if the count is more than fits on one screen. Must have a possibility to search for open multiplayer games in particular genre or by name.

Also listings for multiplayer games that are not in "waiting for more players" state are required to allow users to create their own multiplayer game.

"Join a multiplayer game" and "Start a new multiplayer game".

Developer(s): Kaunisto
Design pair: Pihamaa
Time estimate: 4 hours
Deadline: 17.1.

Multiplay now

Joins in the chosen game and if there is not an available multiplayer game session open, starts a new one and continues to wait for others to join.

Developer(s): Kaunisto
Design pair: Pihamaa
Time estimate: 2 hours
Deadline: 17.1.

Implementing new layout

Change the layout of the developed system to the one created with the assistance of graphics designer.

Developer(s): Auvinen
Design pair: Leiponen
Time estimate: 20 hours
Deadline: 19.1.

Support for skins

The system must support multiple skins on the architectural level. These skins include the possibility to modify the following items:

- The css files used
- The view files used (the html-templates)

The different files reside under sub-directories of each actual directory. The sub-directories should be named according to the skin identifier. E.g.:

```
../css/
       game_styles.css
       user_styles.css
     ../skin_1/
              game_styles.css
     ../skin_2/
             game_styles.css
             user_styles.css
```

The sub-folders will have the same folder-structure as the original folders. If any file is missing from the skin-folder, the one in the default folder is used.

For the view-side functionality to work, the CI view selection functionality must be modified accordingly (see CI documentation on overriding the CI classes).

If a lot of images placed directly into the html are used in the final version, those might have to be separated too, but it is most likely that we will manage without this. Implementing this could prove tricky, since selecting the images in the css-files is hard (no dynamic logic), and the point would be moot without that side also being covered.

The skins must have a database entry, which specifies the name of the skin, the internal identifier (used in the folder names and such) and the type of the skin

The different skin types and the places where they can be selected are:

- system skin
    - application_config.php
- account skin (sales can use this for selecting client-specific skins)
    - admin-side user-management
- profile skin
    - my profile page

No actual alternative skins are created, except for one for testing purposes. The creation of testing purpose skin (does not need to look pretty or differ vastly from the default one) also falls under this task.

Developer(s): Leiponen
Design pair: Jääskeläinen, Auvinen
Time estimate: 8 hours
Deadline: 19.1.

Continuous integration (inc. PHPDoc generation)

Deploy and configure CruiseControl and Ant so that the deployment will happen at least once a day without any human intervention. The individual tasks that should be done in the build cycle are listed below on a generic level:
- Backup the current version
- Overwrite current version with a fresh checkout from SVN
- Run the unit tests
- Run the front-end tests
- If any of the tests fail, revert back to the backup
- Report test results to some appropriate channel and make sure the data will be retained
- Generate PHPDoc documentation

If not possible to do in this way, follow the general intention as specified above as far as possible.

Integrate the automatic PHPDoc generation to the continuous deployment cycle. If not possible, make the generation as easy as possible manually, and ensure there are no complications with this. Send feedback of any code block syntax errors to the respective developers, and either let them fix them or in minor cases fix the errors yourself. Provide the link to the documentation to all developers through Wiki and mail.

Assignee(s): Jääskeläinen
Design pair: -
Time estimate: 6 hours
Deadline: 19.1.

User management (inc. general structure)

The user management is located entirely on the admin side and will use the generic structure and visual components for that side of the system. The user management allows a user with the sufficient rights to create, delete and modify other users she has access to.

The different user levels are:

- Administrator: Access to all users, incl. other administrators.
- Salesman: Ability to create new end-users. Ability to modify and delete users created by herself.
- Account: An end user account for using the actual system

Administrators and salesmen cannot access the end-user system, but must create separate user accounts / profiles for themselves to do that.

The individual profiles are not accessible through this component, only the higher-level user accounts.

The general structure of the admin-side means all the general pieces that must be implemented to enable individual new components to be added to this part of the system later on.

The general structure covers the following issues:

- Separate access url
- Separate login page
- Easily modifiable navigation menu
- Visual layout and sufficient graphics, incl.
    - Generic page layout
    - Navigation menu and other visuals shown on each page
    - Form and content layout and visuals to be used in all the other content on this side

Technical specification:

The individual pieces of data related to the users must include, at the least, the following:

Names
Email
User level
Username
Password
Affiliation (e.g. company)
(Creator, Created, Modified etc.)

The users must be deletable and lockable, incl. an expiry date after which the user will not be able to log in anymore. Deletions must be confirmed by a pop-up or similar method.

Assignee(s): Merinen
Design pair: Jääskeläinen
Time estimate: 10 hours
Deadline: 21.1.

Game session data saving

All of the important data after a gaming session is saved to database. This information should include all possible information such as user(s) profile id(s), game name, game genre, time played and scores for players.

Developer(s): Pihamaa
Design pair: Kaunisto
Time estimate: 5 hours
Deadline: 21.1.

Usability testing

Heuristic testing of the new layout. All workgroup members go through every page of the system using the Nielsen's ten heuristics and writes possible problem areas to report. The reports will be analysed and possible problem areas corrected with the data from the analysis.

Assignee(s): Lehmus, Auvinen, Pihamaa
Time estimate: 10 hours
Deadline: 22.1.

Layout design

Layout design task consist mainly of meetings with user interface consultant. Usually a representative of the customer is also present to discuss about the layout of the system. The design workgroup creates a mock-up of the layout which then will be implemented by the developers.

Assignee(s): Lehmus, Jääskeläinen, Auvinen, Leiponen
Time estimate: 16 hours
Deadline: 23.1.

Buddy list

Every user has a buddy list. The user can add other users on his buddy list (this causes a request to be sent and if it is accepted a buddy is added to buddy list). User sees the online-status of their buddies from the buddy list.

User can invite his buddies to multiplayer rooms and send them internal mails (by means of quick links). These functionalities are not included in this task, though.

The list will be bi-directional (if user A adds user B, A is added to user B's buddy list too). The user must have a way to delete a buddy from his buddy list (also removes it from the ex-buddy's list). The buddy list is accessed from the my profile-page.

Developer(s): Merinen
Design pair: Jääskeläinen
Time estimate: 6 hours
Deadline: 24.1.

Unit tests for old components

Every developer must create unit tests for all the major methods that they have done before, during or immediately after implementing the feature. If the unit tests are found missing during the code reviews, their omission will be communicated to the developer.

Technical specification:

By the fixing freeze points the test coverage will be measured and should be at least 60%, meaning that around 90% of all methods should be tested on at least some level. No negative testing is required, i.e. there is no need to test the results of invalid input except in cases where the input comes from the end user or the likelihood is otherwise high, or if the risks are severe.

The unit test code should be as simple as possible and contain as few external dependencies as is sensible, depending on the particular case. E.g. database methods which contain SQL and access the database can and should be dependent on the database, but a method that uses other methods should not be dependent on them, but use mock objects or other abstractions to circumvent the need.

Developer(s): Everyone
Design pair: Jääskeläinen
Time estimate: 25 hours
Deadline: 24.1.

High scores

Functionalities for recording and viewing high scores for a specific game.

The high-scores must be fetched from the scheduler along with the rest of the external data and stored in the database. The record must contain both a reference to the account & profile ids used as well as the nick used when the game was played (i.e. later changing the nick won't affect these lists).

A page of a game's high scores should be accessible from the game info page. The screen will show the top10, or whatever other number is practical, scores for that game. If at all possible, the different difficulty levels for a game should be separated and the tabs for navigating between those (either tabs or just << >> buttons) should be available on the hi-score screen.

If the data fits into the game info screen opaque layer, it should be placed there. If not, the view will be a totally separate screen. Preferably the first option though.

Some games may not lend themselves well to scoring, e.g. if there are a lot of different maps and the different hi-score categories would thus be too many to be practical. These types of games should, if at all possible, be identified and this feature omitted for those. These will require changes to the meta-data. The developer for this task must plan the required changes early enough for them to be implemented.

Developer(s): Pihamaa
Design pair: Auvinen (Jääskeläinen, Leiponen w/ meta-data)
Time estimate: 10 hours
Deadline: 24.1.

Advertisements

The advertisements and the game slots shown on the front-page must be modifiable from the admin side. Also the distribution of the game slots on the front-page must be modifiable from the admin side.

Regular ads contain the following data:

- Weight: if multiple ads in the same location, what is the weight used for calculating the percentages when one is shown
- Location: where is the ad placed, currently just one position
- URL: where clicking the ad leads to
- Image: upload the gif/jpeg/png image (Are these all supported? What about animated gifs? This must be tested and the input validation and instructions made accordingly.)

There are some special advertisement types:

- Tournament advertisement: One of the open tournaments is selected to be shown here, weights as usual, URL replaced with the link to the tournament info screen.

The different game slot types are:

- Fixed: a certain game is always show here
- Most played: a game from the X most played is randomized here (X is modifiable, 1 for the top 1 played game). If more than 1, no duplicates are allowed to be shown at the same time.
- Personal most played: same as above, but for the current profile
- New game advertisement: The newest game added to the system is shown here

The advertisements must be shown according to the options chosen on the admin side.

Developer(s): Auvinen
Design pair: Pihamaa
Time estimate: 10 hours
Deadline: 26.1.

Sales admin

The sales administration interface will contain the following functionalities:

- Changing the logo (upload, scale, preview)
- Changing the service provider name
- Selecting skin

For the last of these, the developer assigned for the skin task should be consulted. The selection interface will, however, primarily belong under this task.

The use of these elements (logo, company name) on the end-user side according to the selections made also belong under this task.

Developer(s): Leiponen
Design pair: Jääskeläinen
Time estimate: 4 hours
Deadline: 24.1.

Multiplayer lobby

The multiplayer lobby screen is game specific. It lists the open rooms for that game and also displays the buttons for creating new rooms. The list is updated periodically, e.g. every 1 or 2 seconds, or whatever is practical.

The lobby can be accessed from two views: The game info screen and the listing for MP games under all games -> game types -> MP games as well as accessible from the top bar tab "Lobbies".

This task also contains creating the lobbies listing screen, which shows the games that currently have open rooms, as well as all the other games, i.e. the third option above.

These parts will be further designed during the layout design around 22.1. and some of the specifics might yet change.

A full room is probably not shown at all in the listing (mainly due to space constraints). The number of full rooms could anyhow be made visible.
The room creation panel includes the following data:

- room name (should it be unique… depends)
- room size

The updating should preferably be done with some backside asynchronous data transferrals. If that doesn't work due to the limitations of the STBs, either a refreshing iframe or a totally refreshing page can be used.

See the multiplayer rooms-task for some more info regarding those.

Developer(s): Kaunisto
Design pair: Jääskeläinen
Time estimate: 20 hours
Deadline: 28.1.

Multiplayer rooms

The multiplayer rooms allow people to choose who they play with. The room lists the players currently in the room (according to their game-specific nicks or generic nick if game specific not available). It also shows the name of the room as well as the game it is linked to.

The following functionalities are available to the owner of the room:

- kick player
- ban player (cannot enter the same room again)

- start game - this is also accessible even if the number of players has not been reached (even with only 1 player if technically possible)
- modify room name
- modify room size (max no. of players)
- select another game

The following functionalities are available to all members of the room:
- leave room

There are three types of rooms:
- passworded rooms – accessible if password known or through any member's invitation
- invitation only rooms – accessible through any member's invitation
- open rooms – accessible to all

If the owner of the room leaves the room, the next player according to joining order will become the owner. Thus the joining order must be kept track of until at least the starting of the game session.

Developer(s): Kaunisto
Design pair: Leiponen, Jääskeläinen
Time estimate: 20 hours
Deadline: 4.2.

Messaging

The messaging system contains of two parts: fixed messages and e-mail type free content messages.

The fixed messages should contain at least two messages: invitation to a specific multiplayer game/room and request to add the receiver (of the message) to sender's buddy list.

The free content message should work like an internal e-mail but without attachments. These messages can be sent to any user in system, not only those on sender's buddy list but the receiver's global nick name must be known for sending messages to him. A user must have a way to block messages from unwanted sources (for instance in My Profile page: "accept mail only from buddies" or a black list whose mail is automatically rejected).

Incoming invitations are overlaid on the top of the page and can be removed (without a page refresh preferably) by clicking on the decline button. The green and red buttons on the remote control can be used as quick-links for accepting / declining invitations.

Developer(s): Merinen
Design pair: Jääskeläinen
Time estimate: 12 hours

Deadline: 28.1.

Implementing new graphics

Inserting and adjusting the new graphics to the developed system to give it finished look and feel. Includes the effort of adjusting the graphics so that they won't break in different client platforms.

Developer(s): Auvinen
Design pair: Leiponen
Time estimate: 20 hours
Deadline: 29.1.

Setting up demo environment

A task to set up demo environment and to make sure everything works fine at the demonstration.

Developer(s): Kaunisto
Design pair: Leiponen
Time estimate: 4 hours each
Deadlines: 31.1. and 27.2.

Demo to customer

A demonstration session for customer to show visible progress on the project.

Assignee(s): Lehmus, Jääskeläinen, Kaunisto/Leiponen
Time estimate: 6 hours
Deadline: 1.2.

Separate nick for each game

User should be able to specify a different nick name for each game in addition to the global nick used for instance in buddy list. The nick name chosen for a specific game (at the time of the playing) should be showed in the hi-score lists of that game. If no nick name is chosen the global nick will be used.

The nick insertion should be on the game information page. These nicks, as well as the global nicks, are profile-specific. The nick inserted for a game is stored and used also later on for that game.

Developer(s): Merinen
Design pair: Leiponen
Time estimate: 4 hours
Deadline: 2.2.

Usability testing with real graphics

A second round of heuristic testing using Nielsen's heuristics. The same process is used as previously but this time the testing is carried out to make sure that the new graphics will not pose new problems.

Developer(s): Lehmus, Pihamaa, Auvinen
Time estimate: 10 hours
Deadline: 2.2.

Tournaments

There are two kinds of tournaments:
- Hi-score tournaments
- Ladder tournaments

Of these, the first type is simpler and is must be finished first.

Hi-score tournaments:
Hi-score tournaments last a defined period of time. The highs-scores for the selected settings are ordered and shown on a separate page.

Ladder tournaments: DROPPED

This task contains the generic architecture required for represent and save the tournaments, as well as the end-user and admin-side functionalities.

End-user side:
Accessing tournaments (from the game-info screen and via advertisements)
Viewing tournaments info
Viewing tournament results

Admin side:
Creating and modifying tournaments including the following data:
- Game
- Settings
- Selecting the settings for the game
- Timeframe
- Open/closed (if we want to start manually or close before end date)

Tournaments with game session's related to them cannot be deleted altogether, only closed.

Obviously, the setting options for the game must be obtained from the metadata and a game session's settings from the scheduler. Only if the session's settings match the settings selected for the tournament will the game be counted in it.

Because the settings cannot be specified when the game is started, it is up to the player to ensure that the correct settings are used for the game to be counted in the tournament.

Tournament games started when the tournament was open will be counted in the tournament even if the tournament closes during the session.

Developer(s): Pihamaa
Design pair: Jääskeläinen
Time estimate: 25 hours
Deadline: 4.2.

Game filters

Some games can only be shown to some accounts in certain demonstration situations. E.g. the licenses for some games are not valid in the States, thus when the system is used there, those games must be disabled for the demonstration accounts. Also, the filtering of the games allows the salesmen to target the game selection better to suit the customer's needs and to present a more suitable, targeted image (e.g. omit adult games if not a good idea to show them in the demo).

Thus there must be an admin side functionality tied to the account management where the games visible for that account can be selected. The default setting is that all games found from the metadata are available, but by unchecking a checkbox (or something similar) a multi-select box becomes available, where the actual games available for that account are listed (only alphabetically, or should there be some sub-division e.g. by genre).

This task also includes the filtering mechanism which prevents the games not selected from being shown in lists and prevents them from being started. The developer must ensure that the front-page and other listings do not break if a game selected to there is filtered out, although the main responsibility for this resides with the developers of those functionalities.

Developer(s): Leiponen
Design pair: Jääskeläinen
Time estimate: 8 hours
Deadline: 4.2.

Statistics

Functionalities to export statistics information from the data saved from game sessions. The statistics page should be accessible from the admin-side, not from the game system. There should be a way to specify what kind of statistics is wanted for viewing, for instance five most played games (by playing time) for the past three months and average times of playing / logged in.

Developer(s): Merinen

Design pair: Jääskeläinen
Time estimate: 4 hours
Deadline: 4.2.

Static analysis

Run the Zend Code Analyzer for the whole source code at the functional freeze point of the iteration. Divide the results and send them to the respective authors of the code. Make sure the appropriate changes are made and no important warnings are thrown at the fixing freeze point.

Assignee(s): Jääskeläinen
Design pair: Merinen
Time estimate: 1 hours
Deadline: 8.2.

Usability testing with assistance of peer group

This task has two goals. One is to have a cognitive walkthrough with them to find out if there is any serious flaws and the second is to give peer group some training on the use of the system so they can test the ready system more efficiently. The results are reported and analysed. If there is any serious problems, they will be corrected before the system is delivered to peer group for testing.

Assignee(s): Lehmus, Pihamaa, Auvinen
Time estimate: 15 hours
Deadline: 9.2.

Writing test cases

Write test cases for all the major functionalities according to the generic test case design principles. Document the tests in the test matrix excel document or in some other suitable location. The tests should be focused on the most important navigational paths, as the selenium test will cover most of the others. Exploratory testing is used to complement these two. The focus for these test cases should be more on testing other, less visible mid-level functionalities, meaning that the approach is at system and integration level. Such functionalities include e.g. metadata updating.

Assignee(s): Jääskeläinen
Design pair: Merinen
Time estimate: 4 hours
Deadline: 10.2.

Front-end tests

Create a more comprehensive front-end test suite for selenium by the functional freeze point. The tests should cover all the views of the system, incl. the admin side. The most important navigational paths should be covered. This means that the test count should be at least around 15, depending on the size of the individual tests.

Only the basic assertions are needed, including 404 and php errors. Some others might also be good to have included, if deemed appropriate.

The tests must be done within 1 week of the deadline of each functionality and all the tests should be finished by the functional freeze point at 06:00 or by 12:00 during the same day. Short descriptions of the tests cases must be written to the test matrix document containing a listing of the actions and paths and a short descriptive name for the test case.

Assignee(s): Jukka Merinen
Design pair: Juuso Jääskeläinen
Time estimate: 5 hours
Deadline: 10.2.

Billing DROPPED

Billing should be on separate page which is accessible through the admin pages. It should be possible to create exportable billing lists from the gaming data available according to some sets of criteria (time played, number of games played - fixed price, fixed price + time played etc.).

This task will be specified later, after the mid-iteration customer demo.

Developer(s): Jukka Merinen
Design pair: Juuso Jääskeläinen
Time estimate: 5+ hours
Deadline: 11.2.

Credits

The system will have support for an “internal currency”. The credits will be used for playing games. Users get credits assigned to the from the admin side.

This feature is mostly just a demonstration feature at this point. In practice credits could be earned from tournaments or gotten from paying for them. At this point, the idea is only to show the

The amount of credits an account has must be modifiable from the admin-side. The cost for playing each game must be modifiable from the admin-side. The views for these functionalities also belong to this task.

The credit cost for each game must be visible from the game info screen. A nice icon for the credits could be used to represent credits (gold coin, star, or something similar). If a player doesn't have the required credits for starting the game, the starting will be interrupted. The credits are expended upon ending the game. The cost depends on the time spent playing, and is specified as credits/hour. Sessions are not interrupted if the balance goes to zero, as there is no way to do that. The balance can therefore also go negative.

This function should be switchable on/off from the application settings.

Developer(s): Pihamaa
Design pair: Juuso Jääskeläinen
Time estimate: 6 hours
Deadline: 11.2.

Deliver final system to peer group for testing

Create testing guidelines and proper instructions on how to use and access the system. Deliver them to peer group.

Assignee(s): Lehmus, Jääskeläinen
Time estimate: 3 hours
Deadline: 17.2.

Peer group testing, report results

Ad hoc testing of peer group's system in accordance with the testing guidelines provided by them and reporting the results back to the peer group.

Assignee(s): Everyone
Time estimate: 10 hours
Deadline: 21.2

User manual

A thorough user manual will be created. It should contain two parts, one for the end-user (customer) and one for the admin side. User side of the manual contains specific information about using the system and admin side of the manual is about maintaining and configuring the system and users.

Assignee(s): Kaunisto
Time estimate: 15 hours
Deadline: 26.2.

Code reviews

Make code reviews at least twcie for all developers for one of their major code pieces. Cover at least the following points:

- Syntax adherence
- Code structure
- SQL injections
- I18N
- Unit test coverage and quality
- Documentation coverage, syntax conformity and quality

Send the results to the developer and hold a short discussion with each one.

Make more thorough code reviews at the functional freeze point covering at least 80% of the code. Check that the required fixes have been made by the fixing freeze point.

Assignee(s): Jääskeläinen
Time estimate: 10 hours
Deadline: 26.1. & 8.2. + FF 29.1. & 11.2.

System integration to STB

A task to integrate the developed system to the set-top boxes and to solve possible problems which arise for instance from the use of remote control to navigate.

Developer(s): Leiponen, Auvinen
Design pair: Kaunisto
Time estimate: 10 hours

Testing

Run the tests during the week of the deadline of each functionality (Sundays to next week). Report found defects to Eventum. Report unfinished tasks to the project management and to the respective developer. Mark the results in the test matrix.

Run all the tests at the functionality freeze point and once again at the fixing freeze point. Mark the results in the test matrix and create test charters for both. Report found defects to Eventum.

Assignee(s): Jääskeläinen
Design pair: -
Time estimate: 15 hours

Updating documents

Update the following documents
User Requirements Document – Jääskeläinen 2 hours
QA Plan – Jääskeläinen 1 hours
Project Plan – Lehmus 5 hours
Architecture Plan – Kaunisto 3 hours
Coding Convention – Kaunisto 2 hours

Create the following documents

QA report I2 - Jääskeläinen 3 hours
Progress report I2 – Lehmus 9 hours

Assignee(s): Lehmus, Jääskeläinen, Kaunisto
Time estimate: 25 hours
Deadline: 29.12.

4. Resources

Name	**PP**	**I1**	**I1½**	**I2**	**Total**
Ismo Lehmus	41 (50)	54.3 (60)	22.5 (32)	59.5 (68)	177.3 (210)
Juuso Jääskeläinen	38 (50)	69 (60)	25 (51)	79.5 (49)	211.5 (210)
Antti Kaunisto	34 (30)	25 (70)	14 (42)	86 (87)	159 (229)
Joni Leiponen	16 (30)	64 (70)	22 (27,5)	38 (47)	140 (174.5)
Jukka Merinen	9 (10)	59.5 (80)	24 (36,5)	76.2 (63)	168.7 (189.5)
Antti Pihamaa	4.5 (10)	69.9 (80)	28 (20,5)	102.4 (67)	204.8 (177.5)
Tapio Auvinen	6 (10)	57.5 (80)	20.5 (38,5)	104.5 (73)	188.5 (201.5)
Total:	148.5 (190)	399.2 (500)	156 (248)	546.1 (454)	1249.9 (1392)

The hours in columns are actual used hours and the numbers inside parenthesis represent original planned effort. The numbers on columns I2 and Total will be changed as the tasks are completed.